Practical NFC Peer-to-Peer Relay Attack using Mobile Phones

Lishoy Francis, Gerhard Hancke, Keith Mayes, and Konstantinos Markantonakis

Information Security Group, Smart Card Centre,
Royal Holloway University of London,
Egham Hill, TW20 0EX, Surrey, United Kingdom.
(L.Francis;Gerhard.Hancke;Keith.Mayes;K.Markantonakis)@rhul.ac.uk
http://www.scc.rhul.ac.uk/

Abstract. NFC is a standardised technology providing short-range RFID communication channels for mobile devices. Peer-to-peer applications for mobile devices are receiving increased interest and in some cases these services are relying on NFC communication. It has been suggested that NFC systems are particularly vulnerable to relay attacks, and that the attacker's proxy devices could even be implemented using off-the-shelf NFC-enabled devices. This paper describes how a relay attack can be implemented against systems using legitimate peer-to-peer NFC communication by developing and installing suitable MIDlets on the attacker's own NFC-enabled mobile phones. The attack does not need to access secure program memory nor use any code signing, and can use publicly available APIs. We go on to discuss how relay attack countermeasures using device location could be used in the mobile environment. These countermeasures could also be applied to prevent relay attacks on contactless applications using 'passive' NFC on mobile phones.

1 Introduction

Near Field Communication (NFC) [1] is intended as a short-range standardised technology for providing contactless communications for mobile devices. NFC is intended to be an intuitive method of establishing ad-hoc connections, simply requiring that two NFC-enabled devices are brought in close physical proximity to each other. NFC also allows for devices to interact with existing contactless/RFID (Radio Frequency Identification) systems. In 'passive' communication mode NFC allows devices to emulate passive contactless smart cards, while 'active' mode allows for devices to act as contactless smart card readers or to communicate with each other. Although the use of NFC-enabled devices in contactless systems has received much publicity, the use of NFC to support peer-to-peer services is less well covered.

One of the earliest specified uses of active NFC was to pair Bluetooth devices by facilitating the exchange of information needed to setup the Bluetooth

communication channel [2]. NFC can also be used for sharing data and content between mobile devices, such as digital business cards and social networking details, although the data rates are currently not best suited to high-bandwidth transfers. Mobile payments are becoming increasingly popular and there are a variety of schemes using a range of data bearers. NFC is seen as an ideal technology in this area with its ability to interact with existing contactless systems and facilitate peer-to-peer transactions between mobile phones [3].

In short range communication systems, it is usually assumed that the devices are actually in close physical proximity when successfully communicating. However, in a relay attack the communication between two devices are relayed over an extended distance by placing a proxy device within communication range of each legitimate participant and then forwarding the communication using another communication channel. The two legitimate participants receive valid transmissions from each other and therefore assume that they are in close physical proximity. In some systems, especially in smart token environment, this could lead to serious security vulnerabilities [4]. The risk posed to near-field channels by relay attacks and the possibility of using NFC-enabled mobile phones as a relay attack platform have been discussed [5] [6], but a practical relay attack using this platform has not been demonstrated.

In this paper, we describe how a relay attack against peer-to-peer NFC system could be practically implemented. The novelty of this attack is that it also uses available NFC-enabled mobile phones as attack platforms, providing the attacker with off-the-shelf proxy device. The attacker's mobile phones are of acceptable (non-suspicious) form factor, unlike custom built emulators used in other relay attacks. The attack functionality can also be implemented using only software via publicly available APIs in a standard MIDlet (Mobile Information Device Profile or MIDP application) using JSR 118 API [7]. The resources and technical skill required of the attacker are therefore greatly reduced. In Section 2 we provide a brief introduction to NFC communication and relay attacks. We describe the attack implementation in Section 3 and discuss current countermeasures that could be used to mitigate relay attacks in Section 4.

2 Background

NFC technology allows for the integration of contactless technology into active devices, such as mobile phones. NFC operates within the 13.56 MHz Radio Frequency (RF) band and has an operating distance up to 10cm. A NFC-enabled device can act as both a "contactless card" and a "contactless reader". NFC-enabled devices, as specified in ISO-18092/ECMA-340 [1] and ISO-21481/ECMA-352 [8], are compatible with existing contactless systems adhering to ISO 14443 [9], ISO 15693 [10] and FeliCa [11]. The NFC standards also define a communication mode for peer-to-peer (P2P) or 'active' communication, with the purpose of facilitating communication between two NFC-enabled devices. In 'active' NFC, the participants communicate in a "client-server" model. The device that starts the data exchange is known as the Initiator and the recipient is known as the Target. In 'active' mode, the Initiator and Target uses their own generated RF

field to communicate with each other. First the Initiator transmits an RF carrier, which it uses to send data to the Target. Once an acknowledgment for the data sent has been received from Target (by modulating the existing field), the Initiator switches the carrier off. The original Target then reprises the role of Initiator, switching on its carrier, and transmits a response to the original Initiator. For the purposes of reader's clarity, we call the NFC enabled mobile phone configured as the Initiator to be in "writing" mode and the phone configured as the Target to be in "reading" mode.

On NFC-enabled mobile phones the Secure Element (SE) provides the security means to establish trust between service provider and the device. The SE also provides a secure environment for hosting sensitive applications and storing cryptographic keys. Currently there are three main architectures for NFC. The first involves an SE that exists as an 'independent' embedded hardware module, i.e. a stand-alone IC (Integrated Chip) is built into the phone. In the second option, the SE is implemented within the UICC (Universal Integrated Circuit Card) [12]. Of the existing Subscriber Identity Application (SIA) modules such as the Subscriber Identity Module (SIM) [13], Universal Subscriber Identity Module ((U)SIM)[14] and Removable User Identity Module ((R)UIM) [15]. The third option implements the SE on a removable memory component such as a Secure Multi-Media Card (Secure MMC) or Secure Digital card (Secure SD) [16]. The discussion comparing the advantages and disadvantages of the above mentioned architectures is beyond the scope of this paper. It is important to note that the NFC standards does not specify any security services apart from the Signature Record Type Definition [17], leaving the security design to the application developer. The Signature RTD specifies how data is to be signed to ensure data integrity and provide data authentication. This standard is currently being reviewed by the NFC Forum [18].

The handsets that were used in our practical experiments implemented independent SEs, which supported Java Card 2.2.1 [19] (Java Card Open Platform [20]), Global Platform 2.1.1 [21] and Mifare Classic [22] emulation. To implement 'passive' emulation of a contactless token an application must be installed in the secure program memory of the SE. Currently, it is possible to unlock the SE in 'independent' architectures and to install custom applications [23]. However, this would be controlled when the UICC is used as the SE, as access to the UICC is strictly managed by the mobile operator [24]. When implementing 'active' NFC communication the functionality can be entirely controlled via a MIDlet installed in the non-secured application memory of the mobile phone. A developer using the mobile phone as a platform for 'active' communication does therefore not need to gain access to any secure parts of the NFC architecture. The MIDlet can implement the 'active' NFC functionality using standard functions available within the extensions of the public JSR 257 Contactless Communication API [25].

2.1 Relay Attack Theory

The Grand Master Chess problem as discussed in [26] provides a classic example of relay attack. In this scenario a player who does not know the rules of Chess can

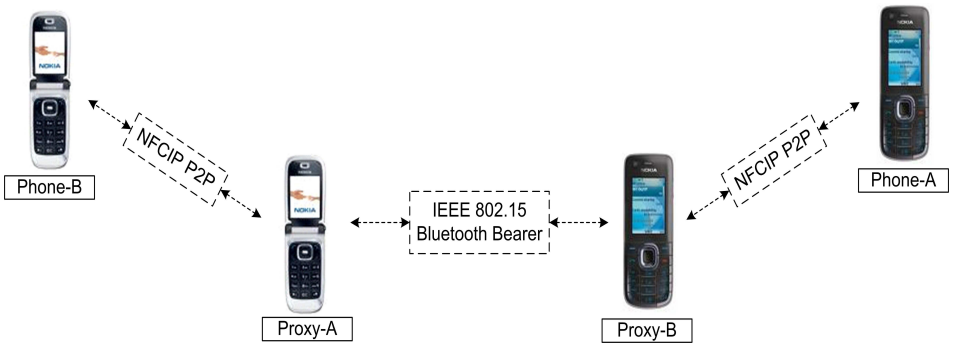


Fig. 1. P2P Relay Setup using Bluetooth.

simultaneously play against two grand masters. The player starts a postal game of Chess with each grand master and subsequently forwards the moves originating from one grand master to the other. Although each grand master thinks that they are engaging the player they are essentially playing against each other. The application of this scenario to security protocols was first discussed in [27] as 'mafia fraud'. Subsequently this attack has also been referred to as a 'wormhole attack' [28] or as a 'relay attack' [29]. Using this attack an attacker is able to bypass security protocols by only relaying challenges and responses between two legitimate entities. As the attacker is always in the position to provide the correct reply, which he/she learned from the other party by forwarding the original message and recording the response, the security protocol is executed successfully and both parties will consider the attacker to be a legitimate participant in the protocol. In this scenario the attacker never needs to know any detail of the information he/she relays, i.e. he/she does not need to know the structure of the protocol, the algorithms used, the plain text data sent or any secret key material. The attacker must only be in a position where he/she can continue to relay the communication for the entire duration of the protocol. Earlier practical examples of relay attack in the contactless environment using custom-built hardware or using NFC-enabled contactless readers can be found in [30], [31] and [32].

3 Relay Implementation

We implemented the relay attack against two NFC enabled mobile phones operating in peer-to-peer mode and participating in a legitimate transaction. As illustrated in the Figure 1, Phone-A intends to interact with Phone-B to perform a legitimate peer-to-peer transaction. The attacker introduces two additional mobile phones into the transaction setup, namely Proxy-A and Proxy-B, to relay the communications between Phone-A and Phone-B. In our proof-of-concept attack experiment, we practically implemented the relay attack using four NFC enabled mobile phones, as shown in Figure 2.

Fig. 2. Devices used in the relay attack – Device B (Nokia 6131 NFC), Proxy A (Nokia 6131 NFC), Proxy B (Nokia 6212 Classic NFC) and Device A (Nokia 6212 Classic NFC).

3.1 Phone-A and Proxy-B

The role of Proxy-B, as the name suggests, is to represent Phone-B and to relay communications to and from Phone-A. To realise both Phone-A and Proxy-B we used two Nokia 6212 Classic NFC mobile phones which are based on Symbian S40 5^{th} Edition FP1 platform. On Phone-A, a MIDlet (MIDP 2.1 application [7]) was implemented (3 kilobytes in size) that utilised the JSR 257 extensions API to realise NFC peer-to-peer communications. Phone-A is designed to switch between "reading" and "writing" modes as required.

On Proxy-B, a MIDlet (MIDP 2.1 application) implemented (14 kilobytes in size) the JSR 257 extensions for NFC peer-to-peer and JSR 82 API [33] for IEEE 802.15 (Bluetooth) communications. By default, Proxy-B was configured in "reading" mode and also supports "writing" mode. The NFC platform of Phone-A and Proxy-B supported the active peer-to-peer mode of operations for both Target and Initiator. Hence these devices performed "reading" and "writing" in active mode.

3.2 Phone-B and Proxy-A

Phone-B and Proxy-A were realised on two Nokia 6131 NFC mobile phones, based on Symbian S40 3^{rd} Edition FP1 platform. Proxy-A represented Phone-A in the transactions and relayed messages with Proxy-B. Similar to Phone-A, on Phone-B a MIDlet (MIDP 2.0 application [7]) was implemented (3 kilobytes in size) that utilised JSR 257 extensions API to realise NFC peer-to-peer communications. Phone-B is designed to switch between "reading" and "writing" modes as required.

On Proxy-A, a MIDlet (MIDP 2.0 application) implemented (14 kilobytes in size) the JSR 257 extensions API [25] for NFC peer-to-peer and JSR 82 API [33]

for IEEE 802.15 (Bluetooth) communications. By default, Proxy-A was configured in "writing" mode and also supported "reading" mode. The NFC platform of Phone-B and Proxy-A supported active peer-to-peer mode of operation for Initiator, but only passive for Target. Hence these devices performed "reading" in passive mode and "writing" in active mode. Using JSR 257 extensions API, the connection open and receiving data in "reading" mode were made as follows,

```
private static final String TARGET_URL
= "nfc:rf;type=nfcip;mode=target";
NFCIPConnection conn = (NFCIPConnection) Connector.open(TARGET_URL);
byte[] data = conn.receive();
byte[] ack = {(byte) 0xFF,(byte) 0xFF};
conn.send(ack);
```

Similarly, the connection open and sending data in "writing" mode were made as follows,

```
private static final String INITIATOR_URL
= "nfc:rf;type=nfcip;mode=initiator";
NFCIPConnection conn = (NFCIPConnection) Connector.open(INITIATOR_URL);
byte[] cmd = {(byte) 0x9A,(byte) 0xED};
conn.send(cmd);
conn.receive();
```

3.3 Relay Bearer

Proxy-A and Proxy-B established the relay channel using Bluetooth, where Proxy-B acted as the "server" and Proxy-A act as the "client". Bluetooth is a short range radio technology developed by Bluetooth Special Interest Group (SIG) and utilises unlicensed radio in the frequency band of 2.45GHz. The supported data speed is approximately 720kps. Bluetooth communication range from 10 metres to 100 metres. The MIDlets implemented the Bluetooth communication on Proxy-A and Proxy-B using JSR 82 API. We used L2CAP (Logical Link Control and Adaption Protocol) available within the host stack of the Bluetooth protocol. L2CAP is layered over the Baseband Protocol and operates at the data link layer in the OSI (Open System Interconnection) Reference Model. It supports data packets of up to 64 kilobytes in length with 672 bytes as the default MTU (Maximum Transmission Unit) and 48 bytes as the minimum mandatory MTU. For creating an L2CAP server connection, a btl2cap scheme (url), a 16 byte service UUID (Universally Unique Identifier), a friendly name (device name) and other parameters were provided as follows,

```
String service_UUID = "00000000000010008000006057028A06";
String url = "btl2cap://localhost:" + service_UUID + ";ReceiveMTU
=672;TransmitMTU=672;name=" + deviceName;
L2CAPConnectionNotifier notifier =
(L2CAPConnectionNotifier) Connector.open(url);
conn = notifier.acceptAndOpen();
```

For creating an L2CAP client connection, a btl2cap scheme, unique Bluetooth MAC address (6 bytes) of the server device, protocol service multiplexer (port) for the remote device and other parameters were provided as follows,

```
String url = "btl2cap://00226567009C:6001;authenticate=false;
encrypt=false;master=false;ReceiveMTU=672;TransmitMTU=672";
conn = (L2CAPConnection) Connector.open(url);
```

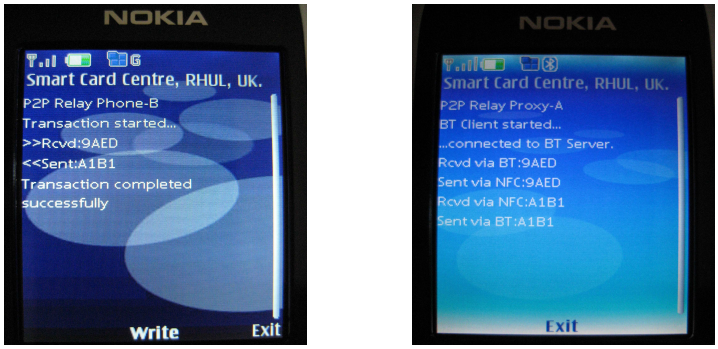
The MIDlets used the DiscoveryAgent class to perform Bluetooth device and service discovery. The DiscoveryListener interface was implemented to handle the notifications of devices and services. When devices and services are discovered, the DiscoveryAgent notifies the MIDlet by invoking callback methods such as deviceDiscovered() and serviceDiscovered(). The methods such as retreivedDevices() and searchServices() caches the previously discovered devices and services, in order to reduce the time needed for discovery. The client MIDlet on Proxy-B listened for the registered service and when available, connected to the server. The sending and receiving of data were achieved by using an L2CAPConnection.

We note that the MIDlets implementing the Bluetooth bearer did not require any code signing [34] and executed in the untrusted 3^{rd} party security domain. More information on Java security domains can be found in [35]. Currently the attacker just needs to enable the application access privilege [36] [37] for connectivity to be set as "always allowed" and enable Bluetooth to be used on the mobile phone. The device specific API access rights can be found in [38] for a Nokia 6131 and in [39] for a Nokia 6212.

Alternatively, Bluetooth can be replaced with SMS (Short Messaging Service) or mobile Internet [using intermediate server(s) via HTTP (Hyper Text Transfer Protocol) over GPRS/E–GPRS (Enhanced General Packet Radio Service)] as the data bearer in order to increase the relay range. These options were only briefly considered, but we found that these bearers introduced additional overheads. In SMS, the user interaction (key press to confirm the message sending) was required even when the MIDlet was signed in the trusted 3^{rd} party security domain. Additionally, SMS is considered to be not so reliable due to its inherent nature such as low bandwidth and variable latency. The additional key presses would also introduce suspicion on the attacker by the victim. Even though, the mobile Internet bearer does not have user interaction overhead similar to SMS, it does need signed code operating within trusted 3^{rd} party security domain.

3.4 Relay Experiment

To demonstrate a proof-of-concept attack, against peer-to-peer NFC transactions using enabled mobile phones, we implemented a simple application in a controlled laboratory environment. Phone-A would act as the Initiator and 'write' a 2-byte message to Phone-B, who would then become the Initiator and answer by "writing" a different 2-byte message back to Phone-A. Our goal was simply to demonstrate that it is possible to relay this exchange or transaction with the mobile-based proxy platforms.



(a) Device B (b) Proxy A

(c) Proxy B (d) Device A

Fig. 3. Relay attack data flow on each device involved.

To start the relay experiment, Proxy-A and Proxy-B negotiates and establishes the Bluetooth channel. In order to simplify the experiment, a 2-byte command message was input by the user in Phone-A configured in "writing" mode. The Phone-A exchanged the command with Proxy-B which then relayed to Proxy-A over the Bluetooth data bearer. Proxy-A being in "writing" mode transfered the payload onto Phone-B. The response for the command message was transferred by Phone-B to Proxy-A in "writing" mode and then relayed over to Proxy-B. The Proxy-B switched to "writing" mode and transfered the response message to Phone-A completing the relay process. In effect, Phone-A was made to believe that the response message to the sent command message was originating from Proxy-B. Similarly, Phone-B was made to believe that command message was originating from Proxy-A, which was actually originating from Phone-A via Proxy-B. Therefore it is reasonable to conclude that, in the real world the security of applications using NFC peer-to-peer communication can be potentially be bypassed using the method presented. The data flow of the relay, as processed by each device, is shown in Figure 3.

4 Using Location Information as Relay Attack Countermeasure

Protecting a system against a relay attack is difficult because this attack circumvents conventional application layer cryptography. Additional security measures are therefore needed to supplement existing authentication or encryption mechanisms. Several countermeasures to relay, or wormhole, attacks have been proposed. These countermeasures are intended for wireless sensor network and smart token environments. Although there are differences between these environments and mobile peer-to-peer services, there are aspects of these countermeasures that could be applied. This section provides a short overview of methods proposed for detecting relay attacks and we refer to an entity verifying the proximity of another entity as the "verifier" and the entity proving its proximity as the "prover".

One way of detecting relay attacks is to monitor any additional delay in the propagation time, which could be caused by the attacker when forwarding the data over a longer distance. Simply placing a time-out constraint on the round-trip time of a challenge-response exchange is not practical as the processing time of the prover can vary and a small error in the expected processing time could have a large influence on the round-trip time [4]. Distance-bounding protocols are designed to enable the verifier to accurately determine the round-trip time of a chosen cryptographic challenge-response [40], thereby eliminating the variability in the time taken to calculate the response. The security of distance-bounding protocols, however, are dependent on the underlying communication channel [41] and conventional channels similar to those used in NFC have been shown to be unsuitable for distance bounding [42]. Furthermore, more computationally advanced devices such as mobile phones can outperform legacy tokens and therefore circumvent systems implementing timing-based countermeasures that allow for processing time of legacy tokens.

The verifier could authenticate the prover based on physical characteristics of the communication channel. Using RF 'fingerprints' has been proposed as a method for source authentication in sensor networks [43] and for smart tokens [44]. An attacker would not be able to relay the communication as his proxy will not have the same RF fingerprint as the prover. This method requires that each device is 'fingerprinted', which is likely to be difficult in practice, especially for a large number of legitimate devices of varying ages (different versions of hardware platform) that are owned, managed or manufactured by multiple entities.

Relay attack could also be mitigated by asking the user to perform additional verification of the transaction. The user could be shown additional details of the underlying transaction by a trusted component to check that the transaction was executed faithfully [45], although this would place a significant responsibility on the user, reduce transaction throughput, and require that he/she has sufficient knowledge of how the transaction should be concluded. Another approach involving the user is multi-channel communication, where the user also verifies additional audio or visual channels [46]. This complicates the relay process, and the attacker must relay multiple channels, some of which might require high

bandwidth. This approach is promising although it might increase the transaction time and reduce simplicity of operation, which are aspects of NFC that are attractive to both users and service providers.

A popular approach in wireless sensor networks is to verify proximity by comparing the location of the nodes [47]. A network-wide localization algorithm is used to determine the relative or absolute location of the nodes. Nodes participating in relay attacks will cause anomalies in the network topology and the attempted attack will be detected. This method usually requires multiple nodes to collaborate in the localization process and constructing a topology, and in the mobile environment a transaction only involved two entities. Location services are, however, becoming prominent in mobile environment providing a basis for implementing a countermeasure that is practical, effective and remains transparent to the end user.

4.1 Integrating Location into NFC Transactions

There are a number of robust Location Based Services (LBS) already deployed in the mobile environment. These services are based on the fact that the location of the handset can be accurately determined, either by the network operator, a third-party or the handset itself. It is therefore feasible that location information could be incorporated into peer-to-peer transactions in order to provide relay-resistant communication. This section briefly discusses two methods for determining handset location that are currently used in LBS.

A simple, and widespread, method of retrieving and handling location information starts with obtaining the information of the cell broadcast tower (or sector) identifier or the Cell-ID, and attaching it with other parameters in the network broadcast such as Mobile Country Code (MCC), Mobile Network Code (MNC) and Location Area Code (LAC). For instance, one can compute a Location Code (LC) as follows,

```
LC=‘‘23415431824422847’’, where MCC=234 | MNC=15 | LAC=43182 | Cell-ID=4422847
```

This information forms the basis for a simple/crude location of the user device, i.e. calculating the position based on the above mentioned location parameters; primarily the tower (or sector) identifier or Cell-ID which the device last accessed. This is applicable to most traditional handsets (e.g. GSM/UMTS). The advanced information obtained from the device can be shown on commercial mapping services, e.g. Google Maps [48]. An important point to note is that the accuracy of this form of deriving the location of a mobile device is very limited and dependent on the radius of the cell coverage, which can range from tens of metres to tens of kilometres. Global Positioning System (GPS) technology is becoming available in an increasing number of mobile handsets and it is a more accurate method of deriving the location information from the mobile phone itself. Providing the handset can detect the satellite signals (which is not always the case), the GPS-enabled handsets supply the application with the location co-ordinates giving a precise position for the device. The co-ordinates consist of Latitude (LAT) and longitude (LNG) information. For example,

```
LAT=51.42869568, LNG=-0.56286722
```

The location information maybe generated by the participants or generated and attested by a trusted $3rd$ party. An example of XML representation of location proof for mobile phones similar to that presented in [49] is given below.

```
<location proof>
<issuer>Issuer's Public Key</issuer>
<recipient>Recipient's Public Key</recipient>
<location information>
<gps><lat>51.42869568</lat><lng>-0.56286722</lng></gps>
<mcc>234</mcc><mnc>15</mnc><lac>43182</lac><cellid>4422847</cellid>
</location information>
</signature></signature>
</location proof>
```

4.2 Preventing Relay Attacks with Location

Examples of enabling mobile applications with 'location proofs' can be found in [49] and in [50]. Assuming that location information is available within mobile phones, the transaction data could be modified to enable the entities involved to verify their proximity. Hu et. al. [51] proposed a simple method using location information to prevent wormhole attacks in wireless sensor networks. This method required that the verifier and prover know their locations and that they have loosely synchronised clocks. The prover would basically add its location and a timestamp to the transmitted data. An additional authentication mechanism, such as a digital signature, is then used to verify that the packet was constructed by the prover. The verifier compares the prover's location to its own and confirms that the prover is in close proximity. If an attacker relays the data then the prover's location should in theory be further away from the verifier's location and the attack would be detected. The attacker cannot modify the data, or construct a new data packet as it does not know the prover's key material. The timestamp prevents an attacker recording a valid transaction and using it at a later stage at the same location. A simple application of this principle is shown in Figure 4. Implementing such a security mechanism in mobile environment is practically feasible and the success of the protocol would simply rely on the accuracy and reliability of the location information available. Using a digital signature scheme, for the required cryptographic data authentication mechanism, is also feasible as mobile phones have sufficient processing resources. The use of digital signatures in NFC applications is also in the process of being standardised in the Signature RTD candidate specification currently being reviewed by the NFC Forum. Also, current work by the authors discusses a secure authentication method for proximity communication channels based on location information. The idea is to attest physical proximity of devices by using location information as an additional security metric. With the identity of the device or subscriber or token bound to the location information it is possible to restrict potential security threats to geographic areas or proximity zones. The method provides a means to test relative and absolute location, and to determine the proximity as well as provide non-repudiation for the devices involved in the transaction.

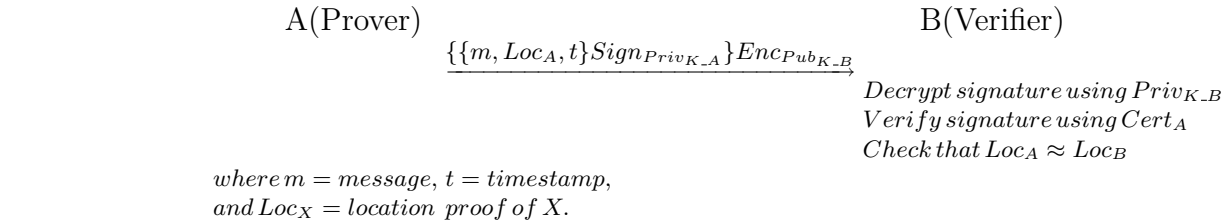


Fig. 4. An example of using location proof for preventing relay attacks in P2P NFC Transactions.

4.3 Limitations of Location Based Security

In this paper, we do not attempt to provide a comprehensive discussion on the issues surrounding the security of location based services, and it should be noted that some location based services have been shown to be insecure, e.g.[52]. However, we do wish to briefly discuss the practical issues in implementing a location-based countermeasure related to the work presented. The limitations of a location-based countermeasure depends upon how the location information is obtained and used within the application. There are external factors, such as host network policy, governing the availability of location information. Some operators may not be prepared to share this information nor to confirm its accuracy unless for legal or investigative reasons. As a result, the application designers would need to consider how the accuracy and detail of available location information could best be exploited. The application design should also take into account the differences in handsets owned by the legitimate participants. The two parties involved in the transaction might subscribe to two different networks. Hence, care needs to be taken in design of the application that generates the location information and its verification, as CellIDs and LACs will vary for different network operator. GPS provides more "independent" access to location information as the mobile phone could determine its own location and not depend on information from the network operator. The downside of this is that the devices would need to support GPS functionality and may not be able to derive GPS co-ordinates indoors. In practice, any application relying on location information to provide security would need to use a combination of location services to best suit the operational environment when a transaction takes place.

5 Conclusion

Peer-to-peer transactions in NFC is being considered for a range of applications including payments. Relay attacks are a threat in contactless and networked environments, and may bypass the security measures employing temporal contracts and cryptography. Our contribution in this paper included a practical demonstration of a first relay attack implementation using NFC-enabled mobile phone platform. We showed that with NFC an attacker can create and introduce proxies by software development (no hardware modification) of suitable MIDlets for the mobile device. The attack did not require any code signing, and did not

need software to be installed in secure program areas such as the SE. It also used standard, easily available APIs such as JSR 257 and JSR 82. The need for countermeasures should therefore be taken seriously, and as discussed the use of location based solution to verify proximity is seen as a promising approach.

References

1. ISO/IEC 18092 (ECMA-340), Information technology Telecommunications and information exchange between systems Near Field Communication Interface and Protocol (NFCIP-1). 2004. http://www.iso.org/, Cited 31 March 2010.
2. Bluetooth Core Specification Version 2.1. + EDR. Volume 2, July 2007.
3. Lin, G., Mikhak, A.A., Nakajima, L.T., Mayo, S.A., Rosenblatt, M.: Peer-to-peer Financial Transaction Devices and Methods. Apple Inc. Patent Application WO/2010/039337, April 2010.
4. Hancke, G.P., Mayes, K.E, Markantonakis, K.: Confidence in Smart Token Proximity: Relay Attacks Revisited. Elsevier Computers & Security, Vol. 28, Issue 7, pp 615–627, October, 2009.
5. Anderson, R.: RFID and the Middleman. Conference on Financial Cryptography and Data Security, pp 46–49, December 2007.
6. Kfir, Z., Wool, A.: Picking Virtual Pockets using Relay Attacks on Contactless Smartcard Systems. Proceedings of IEEE/CreateNet SecureComm, pp 47–58, 2005.
7. Sun Microsystems, JSR-000118 Mobile Information Device Profile 2.0, http://jcp.org/aboutJava/communityprocess/final/jsr118/index.html.
8. ISO/IEC 21481 (ECMA-352), Information technology Telecommunications and information exchange between systems Near Field Communication Interface and Protocol (NFCIP-2). 2005. http://www.iso.org/, Cited 31 March 2010.
9. ISO/IEC 14443, Identification cards – Contactless integrated circuit cards – Proximity cards, http://www.iso.org/, Cited 31 March 2010.
10. ISO/IEC 15693, Identification cards – Contactless integrated circuit cards – Vicinity cards, http://www.iso.org/, Cited 31 March 2010.
11. FeliCa, http://www.sony.net/Products/felica/, Cited 31 March 2010.
12. European Technical Standards Institute (ETSI), Smart Cards; UICC-Terminal interface; Physical and logical characteristics (Release 7), TS 102 221 V7.9.0 (2007-07), http://www.etsi.org/, Cited 31 March 2010.
13. Third Generation Partnership Project, Specification of the Subscriber Identity Module-Mobile Equipment (SIM - ME) interface (Release 1999), TS 11.11 V8.14.0 (2007-06), http://www.3gpp.org/.
14. Third Generation Partnership Project, Characteristics of the Universal Subscriber Identity Module (USIM) application (Release 7), TS 31.102 V7.10.0 (2007-09), http://www.3gpp.org/.
15. Third Generation Partnership Project 2 (3GPP2), Removable User Identity Module (RUIM) for Spread Spectrum Systems, 3GPP2 C.S0023-C V1.0' (May 26 2006), http://www.3gpp2.org/.
16. SD Card Association, http://www.sdcard.org/, Cited 31 March 2010.
17. Candidate Technical Specification: Signature Record Type Definition. NFC Forum. October 2009.
18. Near Field Communication (NFC) Forum, http://www.nfc-forum.org, Cited 31 March 2010.

19. Sun Microsystems, Java Card Platform Specification v2.2.1, http://java.sun.com/products/javacard/specs.html, Cited 31 March 2010.
20. NXP, Java Card Open Platform, http://www.nxp.com/, Cited 31 March 2010.
21. Global Platform, Card Specification v2.1.1, http://www.globalplatform.org, Cited 31 March 2010.
22. NXP Semiconductor. Mifare Standard Specification, http://www.nxp.com/acrobat_download/other/identification/, Cited 31 March 2010.
23. Francis, L., Hancke, G.P., Mayes, K.E., Markantonakis, K.: Potential Misuse of NFC Enabled Mobile Handsets with Embedded Security Elements as Contactless Attack Platforms. Proceedings of the 1st Workshop on RFID Security and Cryptography (RISC'09), in conjunction with the International Conference for Internet Technology and Secured Transactions (ICITST 2009), pp 1-8, November 2009.
24. Mayes, K.E., Markantonakis, K. (Eds.): Smart Cards, Tokens, Security and Applications, Springer Verlag, 2008. ISBN: 978-0-387-72197-2.
25. Sun Microsystems, JSR-000257 Contactless Communication API 1.0, http://jcp.org/aboutJava/communityprocess/final/jsr257/index.html.
26. Conway, J.H.: On Numbers and Games. Academic Press, 1976.
27. Desmedt, Y., Goutier, C., Bengio, S.: Special Uses and Abuses of the Fiat-Shamir Passport Protocol. Advances in Cryptology (CRYPTO), Springer-Verlag LNCS 293, pp 21, 1987.
28. Hu, Y.C., Perrig, A., Johnson, D.B.: Wormhole Attacks in Wireless Networks. IEEE Journal on Selected Areas in Communications (JSAC), pp 370–380, 2006.
29. Hancke, G.P., Kuhn, M.G.: An RFID Distance Bounding Protocol. Proceedings of IEEE/CreateNet SecureComm, pp 67–73, September 2005.
30. Hancke, G.P., Practical Attacks on Proximity Identification Systems (short paper). Proceedings of IEEE Symposium on Security and Privacy, pp 328–333, May, 2006.
31. Libnfc.org, Public Platform Independent Near Field Communication (NFC) Library, http://www.libnfc.org/documentation/examples/nfc-relay, Cited 31 March 2010.
32. RFID IO Tools. rfidiot.org. Cited 31 March 2010.
33. Sun Microsystems, JSR-000082 Java API for Bluetooth 2.1, http://jcp.org/aboutJava/communityprocess/final/jsr082/index.html, Cited 31 March 2010.
34. Sun Microsystems, Java Code Signing for J2ME, http://java.sun.com/, Cited 31 March 2010.
35. Nokia Forum, Java Security Domains, http://wiki.forum.nokia.com/index.php/Java_Security_Domains Cited 31 March 2010.
36. Nokia Forum, MIDP 2.0 API Access Rights, http://wiki.forum.nokia.com/index.php/MIDP_2.0_API_access_rights Cited 31 March 2010.
37. Nokia Forum, MIDP 2.1 API Access Rights, http://wiki.forum.nokia.com/index.php/MIDP_2.1_API_access_rights Cited 31 March 2010.
38. Nokia Forum, Nokia 6131 API Access Rights, http://wiki.forum.nokia.com/index.php/API_access_rights_on_phones,_Series_40_3rd_FP1 Cited 31 March 2010.
39. Nokia Forum, Nokia 6212 API Access Rights, http://wiki.forum.nokia.com/index.php/API_access_rights_on_phones,_Series_40_5th_FP1 Cited 31 March 2010.

40. Brands, S., Chaum, D.: Distance Bounding Protocols. Advances in Cryptology, EUROCYPT '93, Springer-Verlag LNCS 765, pp 344–359, May 1993.
41. Clulow, J., Hancke, G.P., Kuhn, M.G., Moore, T.: So Near and Yet So Far: Distance-Bounding Attacks in Wireless Networks. European Workshop on Security and Privacy in Ad-Hoc and Sensor Networks (ESAS), Springer-Verlag LNCS 4357, pp 83–97, September 2006.
42. Hancke, G.P., Kuhn, M.G.: Attacks on Time-of-Flight Distance Bounding Channels. Proceedings of the First ACM Conference on Wireless Network Security (WISEC'08), pp194–202, March 2008.
43. Rasmussen, K.B., Čapkun, S.: Implications of Radio Fingerprinting on the Security of Sensor Networks. Proceedings of IEEE SecureComm, 2007.
44. Danev, B., Heydt-Benjamin, T.S., Čapkun, S.: Physical-layer Identification of RFID Devices, Proceedings of USENIX Security Symposium, 2009.
45. Anderson, R.J., Bond, M.: The Man-in-the-Middle Defense. Presented at Security Protocols Workshop, March 2006. http://www.cl.cam.ac.uk/ rja14/Papers/Man-in-the-Middle-Defence.pdf
46. Stajano, F., Wong F.L., Christianson, B.: Multichannel Protocols to Prevent Relay Attacks. Conference on Financial Cryptography and Data Security, January 2010.
47. Boukerche, A., Oliveira, H.A.B., Nakamura, E.F., Loureiro, A.A.F.: Secure Localization Algorithms for Wireless Sensor Networks. *IEEE Communications Magazine*, Vol. 46, No. 4, pp 96–101, April 2008.
48. Google Maps, Google Inc., http://www.googlemaps.com/ Cited 31 March 2010.
49. Saroiu, S., Wolman, A.: Enabling New Mobile Applications with Location Proofs. In Proceedings of the 10th Workshop on Mobile Computing Systems and Applications, Santa Cruz, California, February 23 - 24, HotMobile 2009, ACM, New York, USA, 1-6, 2009.
50. Luo, W., Hengartner, U.: Proving your Location without giving up your Privacy. In Proceedings of the Eleventh Workshop on Mobile Computing Systems & Applications, Annapolis, Maryland, February 22 - 23, HotMobile 2010, ACM, New York, USA, 7-12, 2010.
51. Hu, Y.C., Perrig, A., Johnson, D.B.: Packet leashes: A Defense Against Wormhole Attacks in Wireless Networks. Proceedings of INFOCOM, pp 1976–1986, April 2003.
52. Tippenhauer, N.O., Rasmussen, K.B., Pöpper, C., Capkun, S.: Attacks on Public WLAN-based Positioning. Proceedings of the ACM/Usenix International Conference on Mobile Systems, Applications and Services (MobiSys), 2009.

2009 International Conference on Availability, Reliability and Security

Vulnerability Analysis and Attacks on NFC-enabled Mobile Phones

Collin Mulliner
Fraunhofer Institute for Secure Information Technology (SIT)
collin.mulliner@sit.fraunhofer.de

Abstract

Near Field Communication (NFC)-enabled mobile phones and services are starting to appear in the field, yet no attempt was made to analyze the security of NFC-enabled mobile phones. The situation is critical because NFC is mostly used in the area of payment and ticketing. This paper presents our approach to security testing of NFC-enabled mobile phones. Our approach takes into account not only the NFC-subsystem but also software components that can be controlled through the NFC-interface. Through our testing approach, we were able to identify a number of previously unknown vulnerabilities, some of which can be exploited for spoofing of tag content, an NFC-based worm, and for Denial-of-Service attacks. We further show that our findings can be applied to real world NFC-services.

Keywords: *NFC, Mobile Phones, Vulnerability Analysis, Fuzzing, Phishing, Spoofing.*

1. Introduction

Near Field Communication (NFC)-enabled mobile phones and services are starting to appear in the field. According to market research[1], 30% of all sold mobile phones in 2011 will be NFC-enabled. Also, NFC-services heavily rely on the mobile phone network, therefore mobile phone service providers are pushing NFC-enabled phones since these represent an additional source of revenue.

The NFC technology is almost exclusively used for mobile payment and ticketing. Therefore, it is necessary to develop tools and techniques to assess and improve the security of NFC-devices and services.

In this paper, we present our novel approach to the vulnerability analysis of NFC-enabled mobile phones. To the best of our knowledge, no attempt has been made before to analyze or test NFC-enabled mobile phones for vulnerabilities.

We focused our analysis on mobile phones interacting with services that use passive NFC-tags since this is the common setup found in the field today. We acknowledge the fact that future NFC-based payment services will likely be based on smart cards built into every NFC-phone. We however anticipate large scale use of passive tags as soon as NFC-enabled phones hit the critical market penetration.

Vulnerability analysis was partially carried out using fuzzing. Fuzzing was chosen since we did not have access to the source code of the mobile phone. In this work, we present possibilities for fuzzing the NDEF implementation of an NFC-device.

So far, we found several vulnerabilities that allow attacks ranging from spoofing of tag content and interception of tag reading events to Denial-of-Service attacks.

We further present the results of our short survey of NFC-based services. The results verify that the issues discovered through our analysis present a potential threat.

The contributions of this paper are the following:

- We show the steps necessary to perform vulnerability analysis of an NFC-enabled mobile phone, taking into account software components that are not part of the NFC-subsystem but can also be controlled through it.
- We developed a set of tools for security testing of NFC-enabled mobile phones and NFC-services.
- We introduce multiple novel attacks against NFC-enabled mobile phones and services, including a proof-of-concept NFC-based worm.

The rest of this paper is organized as follows: Section 2 provides a brief introduction to Near Field Communication. In Section 3, we discuss related work. In Section 4, we show how we analyzed an NFC-enabled mobile phone and present the issues we found.

1 http://www.abiresearch.com/abiprdisplay.jsp?pressid=719

978-0-7695-3564-7/09 $25.00 © 2009 IEEE
DOI 10.1109/ARES.2009.46

In Section 5, we present our novel attacks against NFC-enabled mobile phones. Section 6 shows that our attacks can be applied to currently deployed NFC-services and in Section 7 we briefly conclude.

2. Near Field Communication

Near Field Communication (NFC) is a bidirectional proximity coupling technology based on the ISO14443 and the FeliCa RFID standards. NFC operates in the 13,56 Mhz spectrum and supports data transfer rates of 106, 216, and 424 kbit/s. The typical communication range of an NFC-device lies between 2 and 4 centimeters. There are three modes of operation for an NFC-device. In *reader/writer* mode an NFC-device acts as a proximity coupling device (PCD). In this mode the NFC-device can read and write data stored on NFC compliant passive transponders. In *card emulation* mode an NFC-device acts as proximity inductive coupling card (PICC). In the *peer-to-peer* mode two NFC devices can carry out bidirectional communication to transfer arbitrary data. Currently deployed services mainly utilize the *reader/write* (PCD) mode, therefore we focused our research around this mode.

2.1. An NFC Mobile Phone

An NFC-enabled mobile phone integrates an NFC-chip such as the NXP PN532[2] and possible a smart card into an ordinary mobile phone.

The NFC-subsystem, if not switched off, is active as long as the mobile phone is ready to accept user input. The NFC-subsystem constantly scans for NFC-tags. As soon as a tag is detected the phone notifies the NFC-aware application that is running. In this case the application has full control and can decide what to do. If a non-NFC-aware or no application is running the operating system reads the tag. If the tag contains data in a supported format the data is passed over to the application that is registered for handling it.

2.2. NFC Data Exchange Format

The NFC Data Exchange Format (NDEF) [4] is a container format used for storing data independent of the tag type. NDEF data is organized in records and messages (a message comprises a set of one or more records). Besides the actual data a record also contains type information about the data, such as a mime-type. NDEF further specifies data types such as the URI [7], Text [6], and Smart Poster [5] record.

A URI can be a simple HTTP URL or more phone centric URI such as `tel` (initiate a voice call) or `sms` (send a predefined short message). A text record contains a human readable string together with a language identifier. The Smart Poster consists of multiple records. In the simplest case these are an `SP` (Smart Poster), a URI, and a Text record. Additional text or image records are possible while there can only be one URI record. This is since the sole purpose of the Smart Poster is to provide additional information about a URI to the human user.

3. Related Work

In [1], Haselsteiner and Breitfuß show multiple attacks against NFC-based systems that rely on the lack of link level security of the NFC technology. The attacks range from eavesdropping to data modification, insertion, corruption, and Man-in-the-Middle attacks.

Rieback et al. show in [3] that RFID systems (NFC is based on RFID) can be attacked like traditional computer systems. They show that RFID-based SQL injection attacks as well as self-replicating RFID viruses are possible. Our NFC-worm uses the same transport medium (a RFID/NFC-tag) but infects NFC-enabled mobile phones rather than RFID middleware.

In [2], Madlmayr et al. give an overview of security measures for NFC use cases and devices. They describe the possibility for NFC-based phishing attacks by simply modifying or replacing NFC-tags. Our work presented in this paper shows attacks that additional abuse vulnerabilities existing in NFC-devices.

4. Analyzing an NFC Mobile Phone

We analyzed the most common NFC-enabled mobile phone, the Nokia 6131 NFC[3]. The phone only supports third party applications created for the Java2 Mobile Edition[4] (J2ME) platform. The user interface of the phone consists of a keypad and a small color display.

We identified NDEF formatted data as the primary input to an NFC-enabled mobile phone acting as tag reader, therefore we focus our analysis on reading, parsing, and displaying of NDEF formatted data. In order to narrow down our analysis we first determined which parts of the NDEF standard are supported by our device. The device seems to support the URI, Text, and the Smart Poster format with the exception of the Action and Image attributes.

Another part of our work was to analyze the NFC J2ME SDK. We present one result of this analysis in Section 5.2.

An important point of our analysis is to test components that are not part of the actual NFC-subsystem but can be controlled through it, for example the web browser and the mobile phone subsystem. These

2 http://www.nxp.com/products/identification/nfc/

3 http://www.forum.nokia.com/devices/6131_NFC

4 http://java.sun.com/javame/

components might have specific issues that are exploitable only, or possibly much easier, through the NFC-interface.

4.1. An NDEF Security Toolkit

For our analysis we have created a toolkit consisting of a flexible NDEF library and a tag reading and writing software for writing the created NDEF data structures to a tag. The tag reading/writing software is based on librfid[5]. All tests described in this paper were carried out using NXP Mifare Classic 1k and 4k RFID tags and a CardMan 5321 RFID reader/writer.

Our NDEF library allows to place arbitrary data into the various NDEF data types such as the URI and Text record. It further allows direct manipulation of length values of various parts of the NDEF format. The reader/writer software does not pre- and post- process any data that is written or read to and from a tag in order to be immune against malformed test data.

Results show that one can easily manipulate the information that is displayed to the user by inserting multiple consecutive whitespace characters into the Text record of a Smart Poster. (These characters cannot be inserted into a Smart Poster through the phone's user interface.) With this kind of manipulation one can prevent the phone from displaying security relevant information to the user or actually making it much harder for the user to locate this information. We will discuss this in more detail in Section 5 where we describe our attacks.

4.2. NDEF Fuzzing

Fuzzing is one of the primary testing methods for software for that no source code is available. Therefore, we also conducted a fuzzing trial against our target device. We only spent little time on fuzzing since even if we could find vulnerabilities, such as buffer overflows, there would be no way to exploit those for code injection as no known code injection techniques exist for the Series40 operating system running on our device.

We identified several fields of the NDEF format that can be utilized for fuzzing-based testing. These are the length fields of the NDEF record payload, type, and ID field. Further we tested the contents of the following fields: payload, type, and ID. We also created test cases for the specific NDEF record types URI and Text, since each consist of multiple fields, specifically the abbreviation in the URI record and the language identifier in the Text record.

The fuzzing process itself is very time consuming since there is almost no automation besides the actual fuzzing data generation. The process has four steps and works as follows:

- The fuzzer generates an NFC/NDEF tag image.
- The image is written to the tag.
- The phone reads the tag.
- The human observes the device for any special behavior such as a sudden reboot.

These four steps are repeated until the fuzzing generator's output space is exhausted or until testing is aborted.

4.2.1. Fuzzing Results We found a bug in the processing of the NDEF record payload length. The two length values `0xFFFFFFFE` and `0xFFFFFFFF` cause the phone to crash and reset. From our observations it seems that only the GUI system crashes since the user is not asked to enter the PIN of the inserted SIM card (this is normally required after a cold boot of the phone). After four crashes in a row the phone automatically powers down.

Further we found that the telephony subsystem that handles the URIs for `tel` and `sms` (voice call and short message, respectively) crashes when given a phone number that consists of exactly 124 characters (numbers 0 to 9). The crash and reset behaves like the one we observed before. We further believe this is an *off-by-one*[6] error since phone numbers with fewer characters are handled correctly and phone numbers with at least 125 characters cause the display of a message box showing an error message.

4.3. The Web Browser

One of the NFC Smart Poster's [5] main purpose is to make the web more accessible to mobile phone users by removing the time consuming and error-prone task of entering URLs like *http://www.example.com/ticket.php?id=12345* using the phone's keypad.

We identified several issues regarding the download of different kinds of files. The most problematic one is that JAR[7] (Java ARchive) files containing application code are downloaded and stored in the application directory as soon as the user activates the browser to open a URL pointing to a JAR file. This situation is critical because the application is executed after the download has completed, also the user still needs to confirm the execution. In comparison, the usual way of installing a J2ME application is to first download the corresponding JAD (Java Application Descriptor)

5 http://openmrtd.org/projects/librfid/

6 http://en.wikipedia.org/wiki/Off-by-one_error

7 http://developers.sun.com/mobility/learn/midp/lifecycle/

file. A JAD contains meta information about the application such as the name, the size, and the URL for the JAR file. After downloading a JAD file the mobile phone usually displays a security warning and provides the user the possibility to inspect the various details of the application. Only after completing these steps an application is installed. In Section 5.2 we show how this behavior can be abused.

Another problem we found is that the web browser does not show the full hostname of a web page being loaded if that hostname is longer than a certain number of characters. Our tests showed that only the last 10 characters of a hostname are displayed. This means a carefully crafted hostname can easily fool the user into believing he is visiting site A while his browser is loading site B.

5. Attacking NFC Mobile Phones

Through our analysis performed on the Smart Poster and URI handling and the results of our fuzzing tests, we derived a number of attacks against NFC-enabled mobile phones.

In the first part we show attacks based on the possibility of spoofing tag content. In the second part we present our proof-of-concept NFC-based worm that spreads using tags. In the final part we briefly discuss the possibility of Denial-of-Service attacks against NFC-enabled mobile phones and services.

5.1. Smart Poster URI Spoofing

In Section 4.1 we discovered a method to influence the display of Smart Poster data through inserting *space*, *tab*, and *new line* characters into the title record. The basic problem lies in the phone's GUI. The Smart Poster is displayed using a fixed sized message box. Inside the message box the title is displayed before the URI. Since the title can be of arbitrary length the URI may not be displayed if the title already uses the whole space provided by the message box. Further since the URI is not marked in any special way, parts of the title can be perceived as being the actual URI.

Although our NFC-phone offers the possibility to further inspect the contents of a Smart Poster it is unlikely that users actually verify the data displayed to them. Further the attacker can even manipulate the inspection-screen to fool the user. This is achieved through moving the actual URI to the second page of the inspection-screen. This is done by adding multiple *new line* characters and a *dot* to the end of the Smart Poster title, shown in Figures 1(b) and 2(b).

5.1.1. Attacking The Web Browser With the possibility to social engineer a victim into loading a malicious website, multiple attacks can be carried out. The most interesting one is a Man-in-the-Middle attack using a web-based proxy. Here the attacker can *steal credentials* or *inject malicious content*.

Attacks like this should work especially well since our test device does not display the URL of the displayed website. This behavior is common among mobile phone web browsers.

In order to try this attack we modified CGIproxy[8] to log all traffic passing through it and added WML [8] support to intercept WAP sites.

Figure 1 shows a simple spoofing attack. Figure 1(a) shows the original tag data as deployed by a service provider while Figure 1(b) shows a spoofed version of the same tag, both Smart Posters look exactly the same when displayed by our test device.

```
Title: Bank of Germany
URL:   https://www.bankofgermany.de
```

(a) Original Smart Poster

```
Title: Bank of Germany\rhttps://www.
         bankofgermany.de\r\r\r\r\r.
URL:   http://www.attacker.com
```

(b) Malicious Smart Poster

Figure 1. URL Spoofing

5.1.2. Attacking The Mobile Telephony Service The possibility of Smart Poster URI spoofing further allows for attacks against the mobile telephony subsystem using the `tel` and `sms` URIs. Here an attacker can craft a Smart Poster that displays phone number A to the user but, when activated, the phone dials phone number B.

Figure 2 shows an example of the telephony URI spoofing attack. Figure 2(a) shows a Smart Poster that would be deployed at a tourist information and Figure 2(b) shows the malicious version of the same poster. A user cannot distinguish one poster from another. The same attack is also possible for predefined short messages (SMS).

The possible impact of such a spoofing attack is high since the victim can be tricked into calling or sending a short message to a premium rate number (for example a 0900 number like shown in Figure 2(b)). These kinds of attacks are likely to happen since the attacker actually has a financial gain out of them.

8 http://www.jmarshall.com/tools/cgiproxy/

```
Title: Tourist Information
URL:   tel:08001234567
```

(a) Original Smart Poster

```
Title: Tourist Information\r080012345
        67\r\r\r\r\r\r\r.
URL:   tel:09009996668
```

(b) Malicious Smart Poster

Figure 2. Telephony URI Spoofing

5.2. A Proof-of-Concept NFC Worm

During our analysis of the NFC-enabled mobile phone and the available functionality of JSR-257 [9] (the standardized NFC Java API) we discovered a possibility to abuse the PushRegistry in order to intercept all URI NDEF messages, NDEF messages that only consist out of a URI record (see the Smart Poster description in Section 2.2 for a counter example). The PushRegistry provides a mechanism for applications to register themselves for handling specific data like, for example, images in the JPEG format. As soon as a tag is read, the PushRegistry will analyze the content of the tag and launch the application registered for the corresponding content. The problem is that the PushRegistry allows an application to register for the generic URI type *urn:nfc:wtk:U*. Therefore, the application is executed for all data types that are based on the URI schema, including the URI type itself. Certain types, such as the Smart Poster, cannot be handled by third party applications.

Being able to intercept all URI NDEF tags we designed and implemented a proof-of-concept NFC-worm that is activated every time a tag of this kind is read. Upon activation the worm tries to spread to the tag by writing a URL that points to a copy of itself (on the Internet) back to the tag. In order to stay undetected, the worm-URL is hidden in a Smart Poster that shows the original URL stored on the tag (using the Smart Poster URI spoofing attack described in 5.1). The next phone reading the tag will get infected by the worm when loading the URL. The user still needs to run the worm binary once, but since he simply needs to confirm the execution after the download, it is likely that the worm is executed.

Infected devices are marked with a cookie that is set while downloading the JAR file containing the worm. If a request to the worm-server contains the cookie, the server answers with a HTTP redirect to the URL originally stored on the tag used by the worm. This is done to keep the tag working and to hide the presence of the worm. The content of a Smart Poster containing the worm-URL is shown in Figure 3.

```
Title: http://example.com/ares09/\r\r\r\r\r\r.
URI:   http://nfcworm.net/worm.php?url=http%3A%2F%2F
         example.com/ares09/
```

Figure 3. NFC-Worm Title and URI

5.3. Denial-of-Service Attacks

Denial-of-Services attacks can be used for destroying the trust relationship between the customer and the service provider. If for example mobile phones crash and reboot every time they touch an NFC-tag, users likely will stop using the service in order to avoid the crash. Such an attack could be implemented by using a sticky paper tag containing a malformed NDEF message and placing it on top of an NFC-tag belonging to the service that is to be discredited. We describe some malformed NDEF messages that cause the phone to crash and reboot in Section 4.2.1.

Users will not be able to link the sticky tag to the crash and therefore will believe that the tag belonging to the service has crashed their mobile phones.

6. Security of NFC-based Services

We conducted a small survey of NFC-based services in order to verify the possible attacks we found. We wanted to determine if the components vulnerable to attacks are used by currently installed services. We only surveyed services that are open to the public and are not in test phase for selected users only.

In order to conduct the survey we have developed a second set of tools to inspect the tags deployed by service operators. The tools run on an NFC-enabled mobile phone in order to ease the survey process.

In the rest of this section we will briefly present the services we surveyed. For each service we will explain how it works and how it can be used in order to attack the users of the service.

All of the services described here use only the built-in functionality of our analyzed NFC-enabled mobile phone, no additional software was installed.

6.1. Wiener Linien

The Wiener Linien is the public transport system for the inner city of Vienna Austria. The NFC-service of the Wiener Linien is a simple e-ticketing system based on short messages (SMS). In order to use the service, a customer just has to hold his phone up against one of the NFC-tags placed next

9 http://jcp.org/en/jsr/detail?id=257

to the ticket vending machine in every subway station. The phone displays the message: `Text Message detected: Für Fahrscheinkauf (Eur 1.70) jetzt senden! +436646606000` (it basically says: to buy a ticket press send). If the user accepts and sends the request, he will receive his ticket as a short message. The fare is deducted from his phone bill or his paybox[10] account.

A possible attack using this service would be replacing NFC-tags with malicious tags that display the same message and phone number using the Smart Poster spoofing attack described in Section 5.1.2, but will instead point to a premium rate phone number owned by the attacker. The victim will likely notice that he did not receive his ticket, but at this point in time the damage is already done. A careful attacker would only replace a single tag in every station in order to trick the user into believing that something just went wrong when buying the ticket and therefore might just try another unmodified NFC-tag.

6.2. Selecta Vending Machines

The Selecta company started installing soda and snack vending machines that offer mobile phone payment using the paybox service. The payment works as follows: each vending machine has a unique identifier in the form of `SNACK257` that is printed on the machine. A customer wishing to buy an item sends a short message containing this identifier to a phone number also printed on the machine. In the next step the machine displays that it is ready to dispense an item. After the customer selected an item the amount is charged to his paybox account. NFC-equipped vending machines feature a tag containing an SMS Smart Poster that contains the same data that is printed on the machine. The customer only needs to read the tag and send the message.

A possible attack on these vending machines could have the goal of buying snacks or soda using somebody else's paybox account. The attack would work as following. The attacker produces a number of fake tags (the vending machine tags are cheap paper tags) that contain the ID of vending machine A. These are mounted on vending machines B, C, and D. The attacker only needs to wait until vending machine A shows that it is ready for selecting an item. This attack has the important advantage that it is nearly untraceable since no premium rate phone number is needed. This attack would again utilize the Smart Poster spoofing vulnerability described in Section 5.1.2.

10 http://www.a1.net/privat/paybox

6.3. Vienna ÖBB Handy-Ticket

The ÖBB Handy-Ticket is a web-based train ticketing system in the city of Vienna Austria. Unfortunately the system was disabled so we could not determine how it actually works. We analyzed the NFC-tags deployed for this system. The tags contain a Smart Poster pointing to a website, the URLs look like this: `http://live.a1.net/oebbticket?start=Wien%20Mitte&n=2`.

A Man-in-the-Middle attack as described in Section 5.1.1 would be the most likely attack since here, an attacker could steal user credentials and/or inject malicious content such as a link to a piece of malware like our proof-of-concept NFC-worm (see Section 5.2)

7. Conclusions and Future Work

We presented a novel method to perform vulnerability analysis of NFC-enabled mobile phones through the application of fuzzing using NFC-tags. We not only analyzed the NFC-subsystem but also components that can be controlled through the NFC-interface, such as the web browser. The testing tools developed for our analysis are freely available from our website.

Through our analysis we found several security vulnerabilities in an NFC-enabled mobile phone of which some can be abused for phishing, an NFC-based worm, and Denial-of-Service attacks.

Future work includes improving the fuzzing process through automation. Also future NFC-devices are likely to be more complex, efficient methods have to be developed to analyze these.

References

[1] E. Haselsteiner, K. Breitfuß. Security in Near Field Communication (NFC). In *Workshop on RFID Security*, 2006.

[2] G. Madlmayr, J. Langer, C. Kantner, J. Scharinger. NFC Devices: Security and Privacy. In *Third International Conference on Availability, Reliability and Security*, pages 642–647, 2008.

[3] M. R. Rieback, B. Crispo, A. S. Tanenbaum. Is Your Cat Infected with a Computer Virus? In *Fourth IEEE International Conference on Pervasive Computing and Communications PerCom*, pages 169–179, 2006.

[4] NFC Forum. NFC Data Exchange Format (NDEF). `http://www.nfc-forum.org`.

[5] NFC Forum. Smart Poster Record Type Definition. `http://www.nfc-forum.org`.

[6] NFC Forum. Text Record Type Definition. `http://www.nfc-forum.org`.

[7] NFC Forum. URI Record Type Definition. `http://www.nfc-forum.org`.

[8] WAP Forum. Wireless Application Protocol Wireless Markup Language Specification. `http://www.wapforum.org`.